OOP With Java

Roll no. 248121

Name: Akshay Chavan

Lab Assignment 5

Code:

FruitBasket tester:

package com.app.tester;

import java.util.Scanner;

import com.app.fruits.Apple;

import com.app.fruits.Fruit;

import com.app.fruits.Mango;

import com.app.fruits.Orange;

public class FruitBasket {

public static void main(String[] args) {

try(Scanner sc = new Scanner(System.in)) {

System.out.println("FruitBasket Application \n Enter basket size: ");

Fruit[] basket = new Fruit[sc.nextInt()];

int counter = 0;

boolean exit = false;

while(!exit) {

System.out.println("1. Add mango\n" + "2. Add Orange\n" + "3. Add Apple\n" + "4. Display all fruits in basket\n" + "5. Display name, color, weight and taste\n" + "6.Invoke fruit specific fxnality\n" + "0. Exit\n" );

System.out.println("Enter choice: ");

switch(sc.nextInt()) {

case 1 :

if(counter < basket.length) {

System.out.println("Enter mango weight");

basket[counter++] = new Mango("Mango", "Yellow", sc.nextDouble() , true);

} else

System.out.println("Basket Full!");

break;

case 2:

if(counter < basket.length) {

System.out.println("Enter Orange weight");

basket[counter++] = new Orange("Orange", "orange", sc.nextDouble() , true);

} else

System.out.println("Basket Full!");

break;

case 3:

if(counter < basket.length) {

System.out.println("Enter Apple weight");

basket[counter++] = new Apple("Apple", "Red", sc.nextDouble() , true);

} else

System.out.println("Basket Full!");

break;

case 4:

for(Fruit fruit : basket) {

if(fruit != null ) {

System.out.println(fruit.toString()); //handled by upcasting - is implicit - is done by compiler

}

}

break;

case 5:

for(Fruit fruit : basket) {

if(fruit != null && fruit.isFresh()) {

System.out.println(fruit.taste()); //runtime poly

}

}

break;

case 6:

//we wanna access subclass specific fxn here, but since our ref var is of superclass type, and objs are created at runtime, compiler has no access/cant identify those fxns

//so need to downcast here

System.out.println("Enter index: ");

Fruit f = basket[sc.nextInt()-1];

if (f != null) {

if(f instanceof Apple)

((Apple)f).jam();

if (f instanceof Mango)

((Mango)f).pulp();

if (f instanceof Orange)

((Orange)f).juice();

}

// for (Fruit fruit : basket) {

// if(fruit != null ) {

// if (fruit instanceof Apple) {

// ((Apple)fruit).jam();

// }

// if (fruit instanceof Mango) {

// ((Mango)fruit).pulp();

// }

// if (fruit instanceof Orange) {

// ((Orange)fruit).juice();

// }

// }

// }

break;

case 0:

exit = true;

break;

}

}

}

}

}

Fruit superclass

**package** com.app.fruits;

**public** **class** Fruit {

**private** String name, color;

**private** **double** weight;

**private** **boolean** isFresh;

**public** Fruit(String name, String color, **double** weight, **boolean** isFresh) {

**this**.name = name;

**this**.color = color;

**this**.weight = weight;

**this**.isFresh = isFresh;

}

@Override

**public** String toString() {

**return** "Name " + name + " Color " + color + " weight " + weight;

}

//taste method

**public** String taste() {

**return** **this**.toString() + "\nNo specific taste";

}

//getters and setters

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **double** getWeight() {

**return** weight;

}

**public** **void** setWeight(**double** weight) {

**this**.weight = weight;

}

**public** **boolean** isFresh() {

**return** isFresh;

}

**public** **void** setFresh(**boolean** isFresh) {

**this**.isFresh = isFresh;

}

**public** **void** setColor(String color) {

**this**.color = color;

}

**public** String getColor() {

**return** color;

}

}

Apple

**package** com.app.fruits;

**public** **class** Apple **extends** Fruit {

**public** Apple(String name, String color, **double** weight, **boolean** isFresh) {

**super**(name, color, weight, isFresh);

}

//overriding taste method

@Override

**public** String taste() {

**return** **this**.toString() + " Taste - Sweet and sour";

}

//subclass specific fxnality

**public** **void** jam() {

System.***out***.println(**this**.getName() + " Weight " + **this**.getWeight() + "\n Making jam!");

}

}

Mango

**package** com.app.fruits;

**public** **class** Mango **extends** Fruit {

**public** Mango(String name, String color, **double** weight, **boolean** isFresh) {

**super**(name, color, weight, isFresh);

}

//Override taste method

@Override

**public** String taste() {

**return** **this**.toString() + " Taste Sweet";

}

//specific fxnality

**public** **void** pulp() {

System.***out***.println(**this**.getName() + " Weight " + **this**.getWeight() + "\n Creating pulp of Apple!");

}

}

Orange

**package** com.app.fruits;

**public** **class** Orange **extends** Fruit {

**public** Orange(String name, String color, **double** weight, **boolean** isFresh) {

**super**(name, color, weight, isFresh);

}

//Override taste

@Override

**public** String taste() {

**return** **this**.toString() + " Taste sour";

}

//subclass specific fxnality

**public** **void** juice() {

System.***out***.println(**this**.getName() + " Weight " + **this**.getWeight() + "\n Extracting juice!");

}

}

Output:







